CPE301 – SPRING 2019

Design Assignment 2C

Student Name: Moriah Wingrove

Student #: 5000156247

Student Email: Wingrove@unlv.nevada.edu

Primary Github address: <https://github.com/windew/Tiny_Dragons.git>

Directory:

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

No components were used for design assignment 2C due to not having the microcontroller and necessary parts from the mail or put together.

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 2/C TASK1 2A.1**

**Asm code**

;

; Design\_Assignment\_2C\_asm.asm

;

; Created: 10/10/2019 9:56:13 PM

; Author : Moriah Wingrove

;

; Goal: Use GPIO and delays using timers and interupts:

;TASK 1: Implement design assignment 2A using Timer 0 - normal mode. Count OVF occurrence if needed. Do not use interrupts

;2A was to generate a waveform on portb.3 with 40% DC and 0.625s period

begin:

sbi ddrb,3 ;pb3 is an output

cbi portb, 3 ;Led is off

ldi r23, 0x61

timer:

ldi r20, 0x0f ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r22, 0x00

out tccr0a, r22 ; timer0, normal

ldi r21, (1<<CS02) ;

out TCCR0B, r21; send r21 out to timer

check:

in r21, TIFR0 ;read overflow

sbrs r21, 0; skip if overflow occurs

jmp check

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r23

brne timer

blink:

ldi r23,0x41

sbi portb, 3 ; turn on LED on Portb bit 3

timer2:

ldi r20, 0x0f ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r22, 0x00

out tccr0a, r22 ; timer0, normal

ldi r21, (1<<CS02) ; prescalar 256

out TCCR0B, r21; send r21 out to timer

check2:

in r21, TIFR0 ;read overflow

sbrs r21, 0; skip if overflow occurs

jmp check2

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r23

brne timer2

rjmp begin ; loop back to the start

**C code**

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/delay.h>

/\*

\* DA2C\_1\_C.c

\*

\* Created: 10/13/2019 12:42:27 AM

\* Author : Moriah Wingrove

\*/

#include <avr/io.h>

int main(void)

{

unsigned char count = 0; //count to keep track of timer0

DDRB |= (1<<3); //portb.3 output of LED

PORTB = 0; // led is off

while (1)

{

//set timer0 with prescalar 256 and normal mode

TCCR0A = 0;

TCCR0B |= (1<<CS02);

TCNT0 = 0; //counter initiallized

while((TIFR0 & 0X1) == 0); //WAIT for overflow

TCCR0B = 0; //stop timer

TIFR0 |= 1; //clear overflow

if (count > 90) //edit

{

PORTB |= (1<<3); //led is on

if(count < 152 )

{

count++;

}

else

{

count = 0;

PORTB = 0; //led is off

count++;

}

}

else

count++;

}

}

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 2/C TASK1 2A.2**

;

; Design\_Assignment\_2C\_T1\_2A2\_asm.asm

;

; Created: 10/14/2019 12:14:14 AM

; Author : Moriah Wingrove

;

;TASK 1: Implement design assignment 2A using Timer 0 - normal mode. Count OVF occurrence if needed. Do not use interrupts

;2A Connect a switch to PORTC.3 (active high - turn on the pull up transistor)

;to poll for an event to turn on the led at PORTB.2 for 1.333 sec after the event.

begin:

cbi DDRC, 3 ;sets port c as input

sbi portc, 3 ; sets port c bit 2 high to enable pull up

sbi ddrb, 2 ;sets port b as output

cbi portb, 2 ;LED is off

poll\_loop:

sbic pinc, 3 ;if pinc is low (0) skip next instruction

jmp LED ;jumps to label led

sbi portb, 2 ;turn on led

ldi r23, 0x5a

ldi r24, 0x02

timer:

ldi r20, 0x0f ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r22, 0x00

out tccr0a, r22 ; timer0, normal

ldi r21, (1<<CS02) ;

out TCCR0B, r21; send r21 out to timer

check:

in r21, TIFR0 ;read overflow

sbrs r21, 0; skip if overflow occurs

jmp check

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r23

brne timer

dec r24

brne timer

LED:

cbi portb, 2 ;turns led off

jmp poll\_loop ;go to label poll\_loop

**C code**

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/delay.h>

/\*

\* DA2C\_1\_C.c

\*

\* Created: 10/13/2019 12:42:27 AM

\* Author : Moriah Wingrove

\*/

#include <avr/io.h>

int main(void)

{

unsigned char count = 0; //count to keep track of timer0

DDRB |= (1<<3); //portb.3 output of LED

PORTB = 0; // led is off

while (1)

{

//set timer0 with prescalar 256 and normal mode

TCCR0A = 0;

TCCR0B |= (1<<CS02);

TCNT0 = 0; //counter initiallized

while((TIFR0 & 0X1) == 0); //WAIT for overflow

TCCR0B = 0; //stop timer

TIFR0 |= 1; //clear overflow

if (count > 90) //edit

{

PORTB |= (1<<3); //led is on

if(count < 152 )

{

count++;

}

else

{

count = 0;

PORTB = 0; //led is off

count++;

}

}

else

count++;

}

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/C from TASK 2**

**Asm code**

;

; Design\_Assignment\_2C\_T2\_asm.asm

;

; Created: 10/13/2019 1:37:35 AM

; Author : Moriah

;

;TASK 2: Implement design assignment 2A using Timer0\_OVF\_vect interrupt mechanism in normal mode.

;2A was to generate a waveform on portb.3 with 40% DC and 0.625s period

.org 0 ;location of main prgm

jmp main ;go to main prgm

.org 0x20 ;location on interrupt

jmp T0\_OVF ;jmp to interrupt

main:

sbi ddrb,3 ;pb3 is an output

cbi portb, 3 ;Led is off

ldi r23, 0x58

timer:

ldi r20, 0x0 ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r22, 0x00

out tccr0a, r22 ; timer0, normal

ldi r21, (1<<CS02) |(1<<CS00) ;

out TCCR0B, r21; send r21 out to timer

;enable interrupts

ldi r20, 0x01 ;r20 =1

sts TIMSK0, r20; store r20 in TIMSKO - enables interrupt overflow

sei ; interrupts enabled

wait:

jmp wait ;loop until interrupt occurs

T0\_OVF:

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r23 ;timer needs to overflow

brne done

sbi portb, 3 ;turn on led

ldi r23, 0x3e

timeron:

ldi r20, 0x00 ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r21, (1<<CS02) ;

out TCCR0B, r21; send r21 out to timer

check:

in r21, TIFR0 ;read overflow

sbrs r21, 0; skip if overflow occurs

jmp check

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r23

brne timeron

rjmp main

done:

ldi r20, 0x00 ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r21, (1<<CS02) ;

out TCCR0B, r21; send r21 out to timer

reti

**C code**

/\*

\* design\_assignment\_2C\_T2\_c.c

\*

\* Created: 10/13/2019 1:48:29 PM

\* Author : Moriah Wingrove

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <avr/io.h>

#include <avr/interrupt.h>

// 1Mhz clock & 64 prescaler

// global variable for keeping track of # of times Timer0 overflows

volatile int count;

// this interrupt service routine (ISR) runs whenever an overflow on Timer0 occurs

int main(void) {

count = 0; // initialize count to keep track of number of

DDRB |= (1 << 3); // connect LED to pin PB.2

PORTB = 0; //led is off

// set up Timer0 with prescaler = 1024 and normal mode

TCCR0A = 0;

TCCR0B |= (1 << CS02)|(1 << CS00);

TCNT0 = 0; // initialize counter

TIMSK0 |= (1 << TOIE0); // enable overflow interrupt

sei(); // enable global interrupts

while(1) ; // loop forever

}

ISR (TIMER0\_OVF\_vect)

{

if (count > 21)

{

PORTB |= (1 << 3); // Toggle PB.3

if(count < 37 )

{

count++;

}

else

{

count = 0;

PORTB = 0;

count++; // reinitialize count

}

}

else count++;

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/C from TASK 3**

**Asm code**

;

; Design\_Assignment\_2C\_T3\_asm.asm

;

; Created: 10/13/2019 3:29:25 AM

; Author : Moriah Wingrove

;

.org 0x00 ;reset location

jmp main

.org 0x1c ;location for timer0 cmpA

jmp t0\_cm\_isr

main:

ldi r23, 23 ;MAKE 23 FOR 375 DELAY

ldi r24,0x3e

cbi portb,3 ;led off

timer:

OUT SPH,R20

LDI R20,LOW(RAMEND)

OUT SPL,R20

LDI R20,0xff

OUT OCR0A,R20 ;OCR0A

LDI R20,0x02 ;(1<<WGM01)

OUT OCR0B,R20 ;Timer in CTC mode

; Force Output Compare A

LDI R20,0x0d ;

OUT TCCR0B,R20 ; Force Output & Start Timer0

SBI DDRB,3 ;PB3 as an output

LDI R20,(1<<OCIE0A) ;Timer0 compare match

sts TIMSK0, r20

SEI ;Set I – SET Main Interrupt Enable after all config

wait: JMP wait

;--------ISR for Timer 0

t0\_cm\_isr:

LDI R20, (1<<OCF0A)

OUT TIFR0, R20

dec r23 ;timer needs to overflow

brne timer

sbi portb,3

timeron:

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

ldi r20, 0x00 ;r20 = 0

out tcnt0, r20 ;load timer 0

ldi r21, (1<<CS02) ;

out TCCR0B, r21; send r21 out to timer

check:

in r21, TIFR0 ;read overflow

sbrs r21, 0; skip if overflow occurs

jmp check

out tccr0b, r22 ; stop timer

ldi r21, (1<<tov0)

out TIFR0, r21 ;clear overflow

dec r24

brne timeron

rjmp main

1. **SCHEMATICS**

No schematics due to lack of microcontroller

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

TASK 2C\_T1\_2A.1:

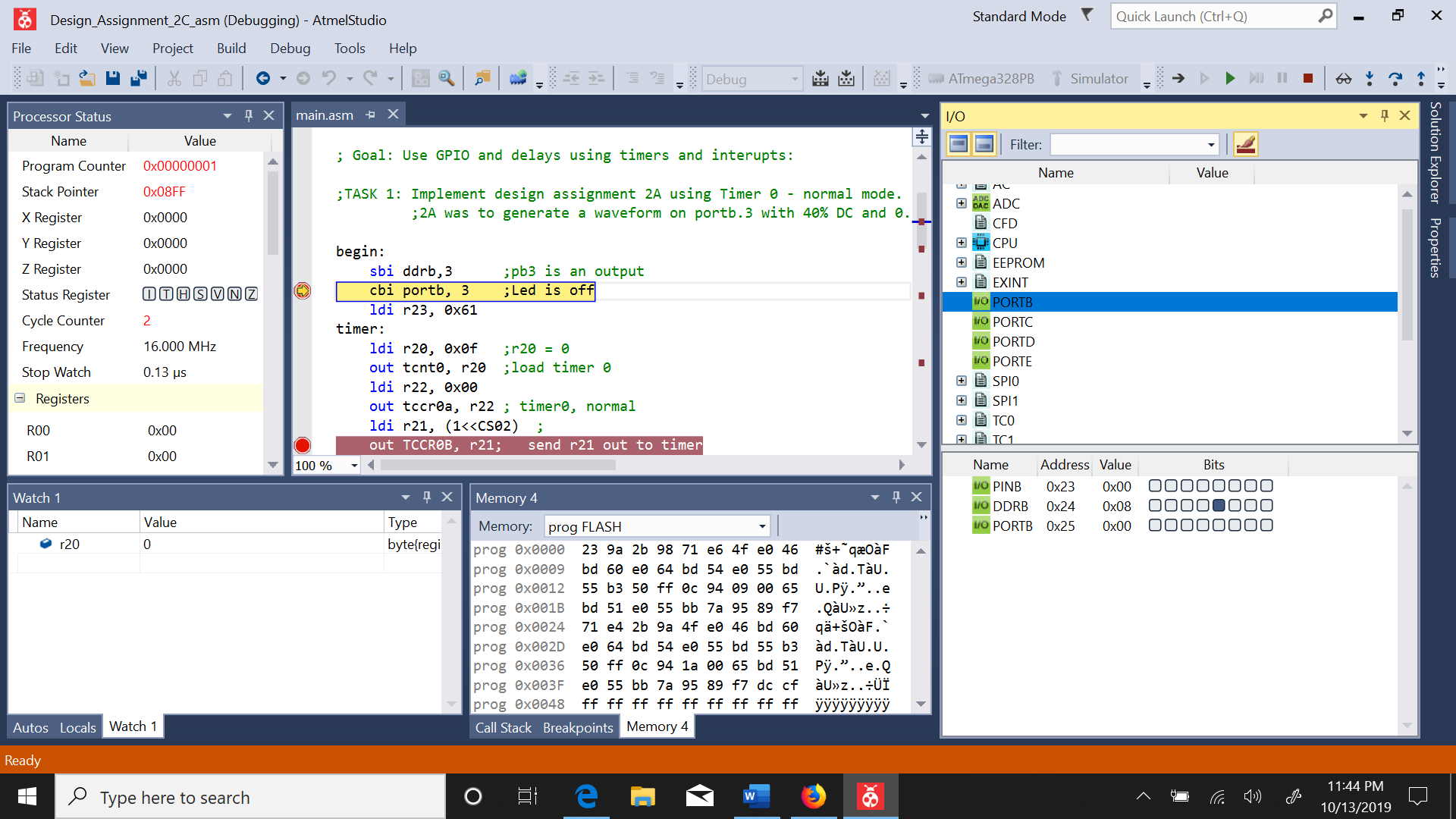


Figure : before portb.3 is set watch is at 0.13us

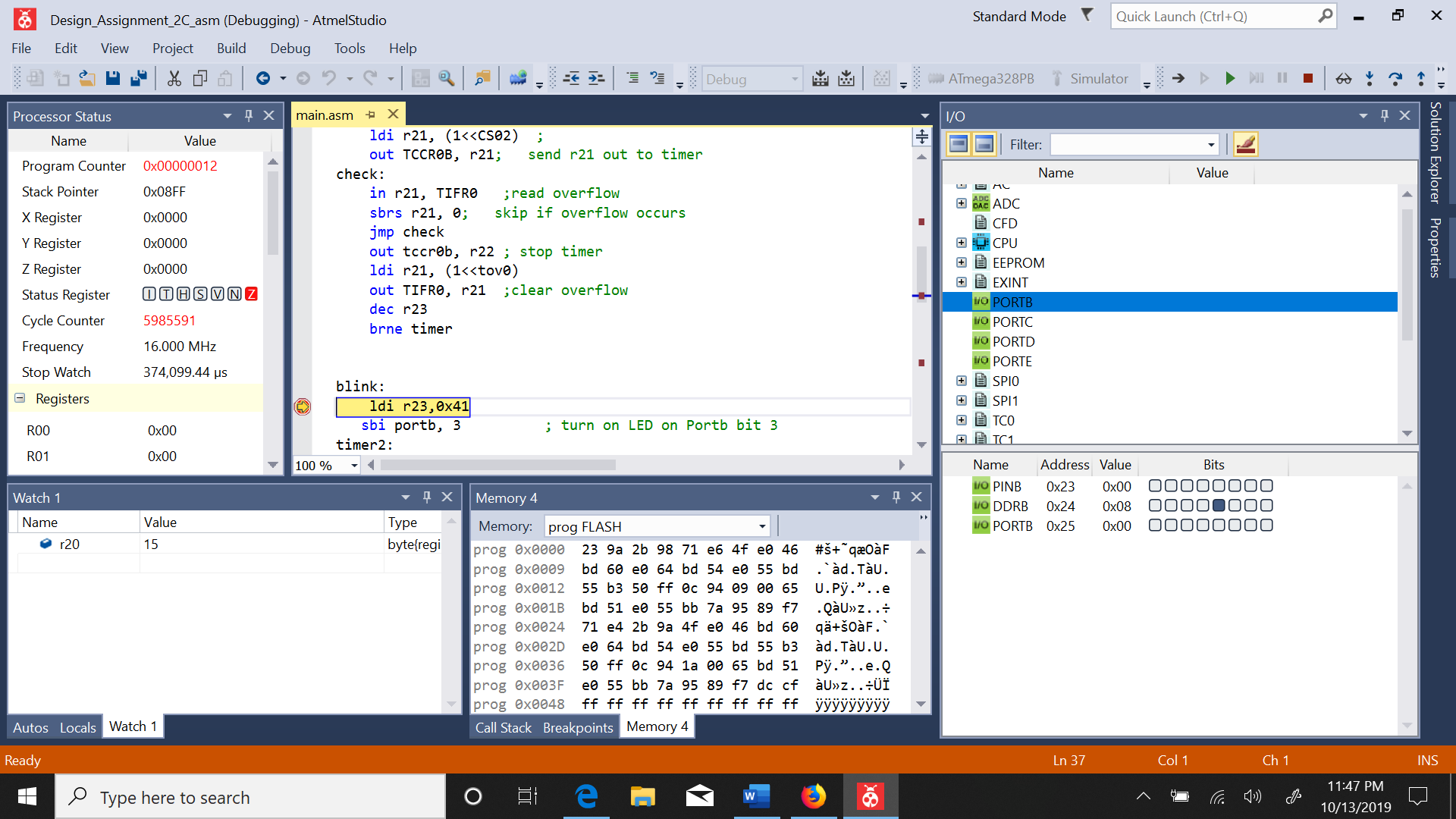


Figure : led is off for 374ms

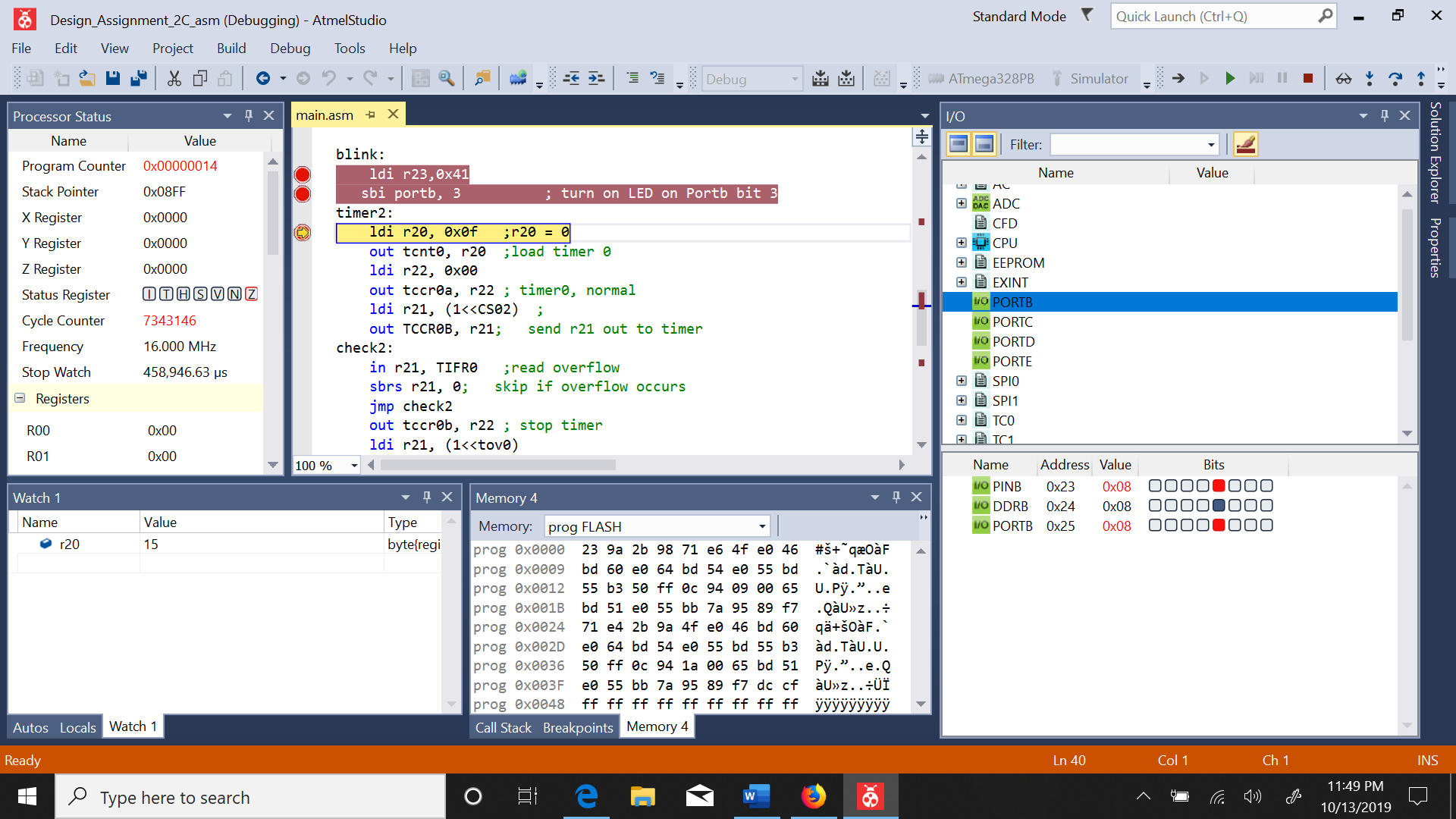


Figure : portb3 is set led is on

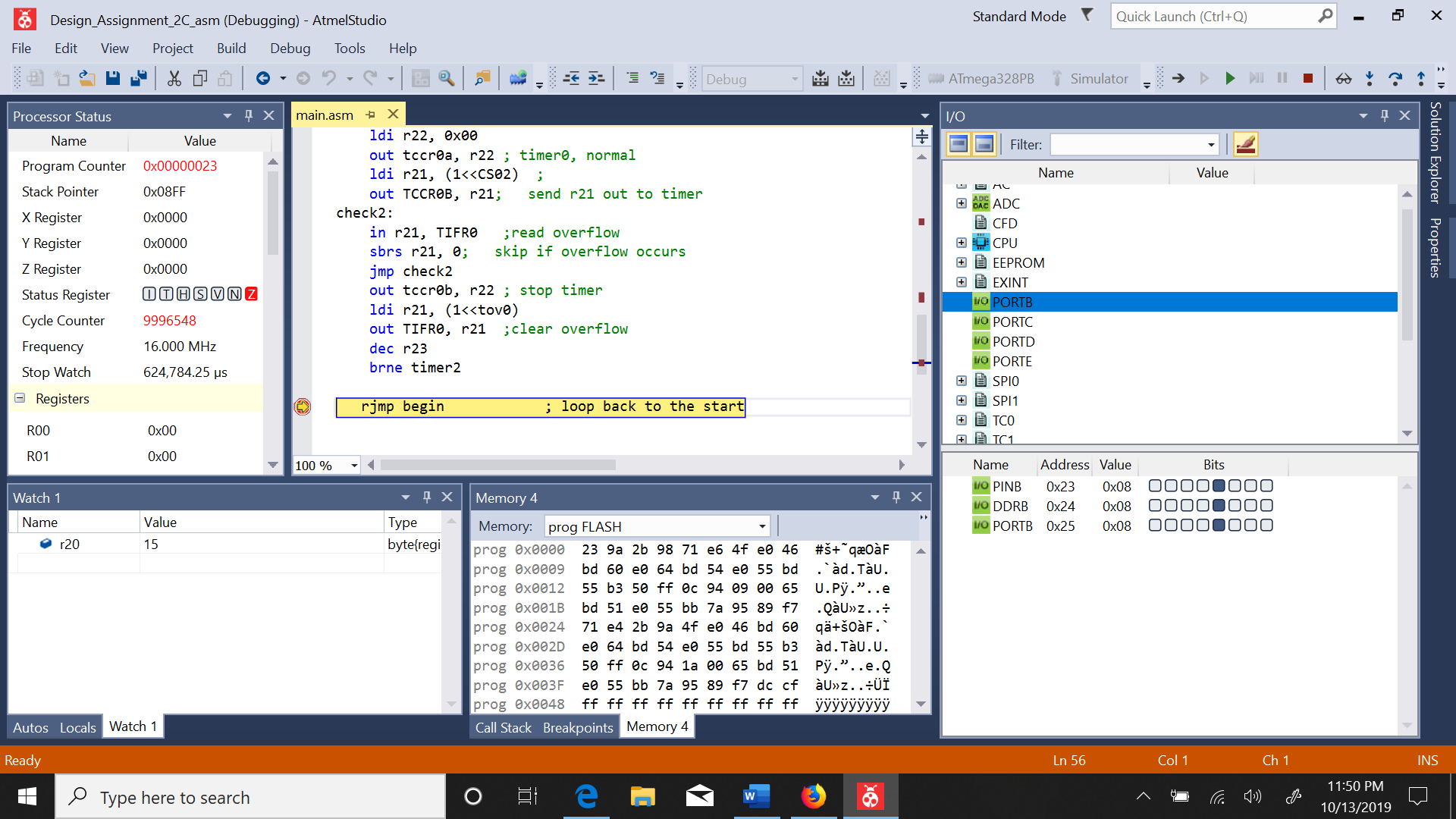


Figure : The led has been turned on for 250ms and total period is 625ms

TASK 2C\_T1\_2A.2:

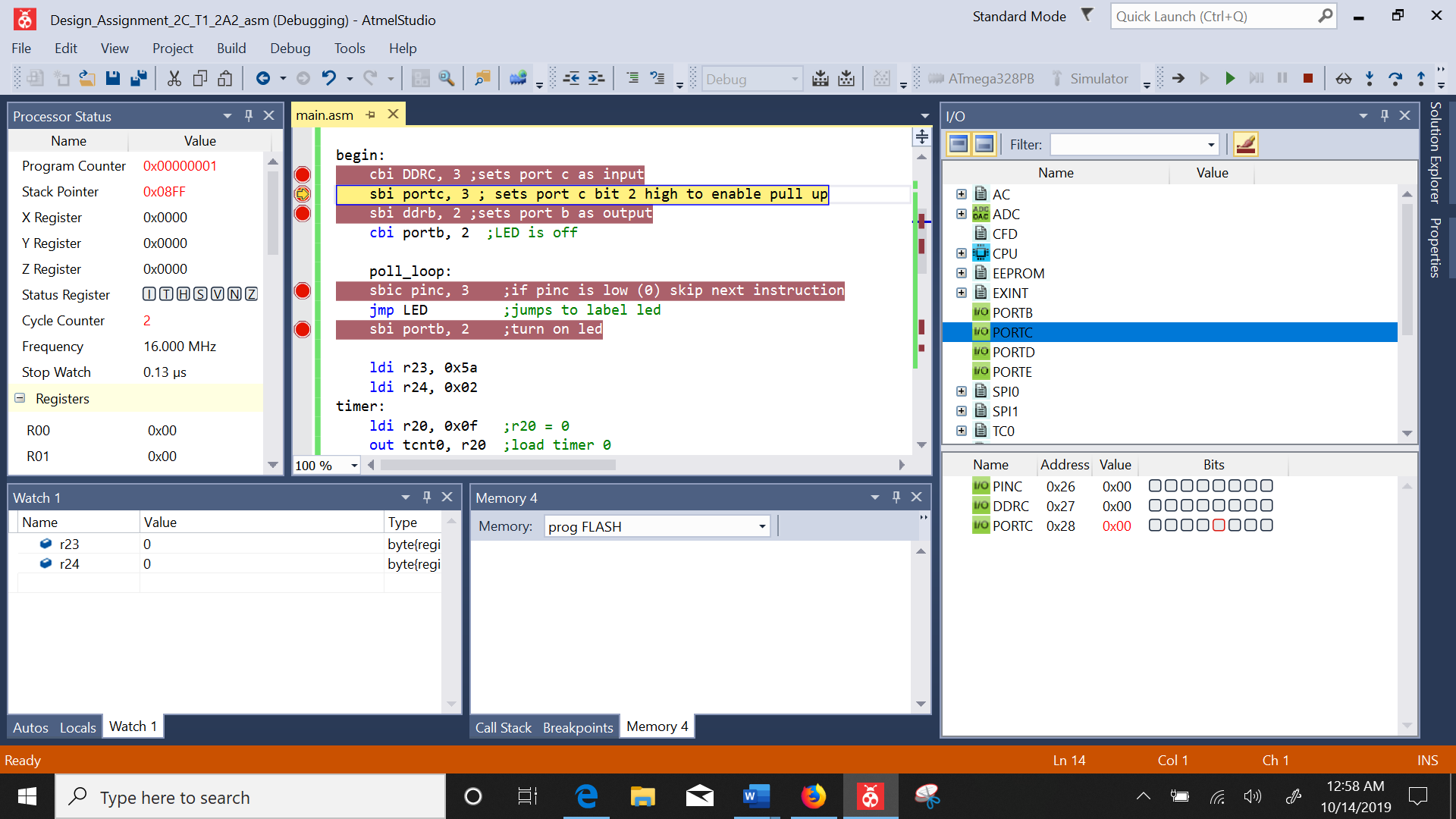


Figure :Before code is debugged

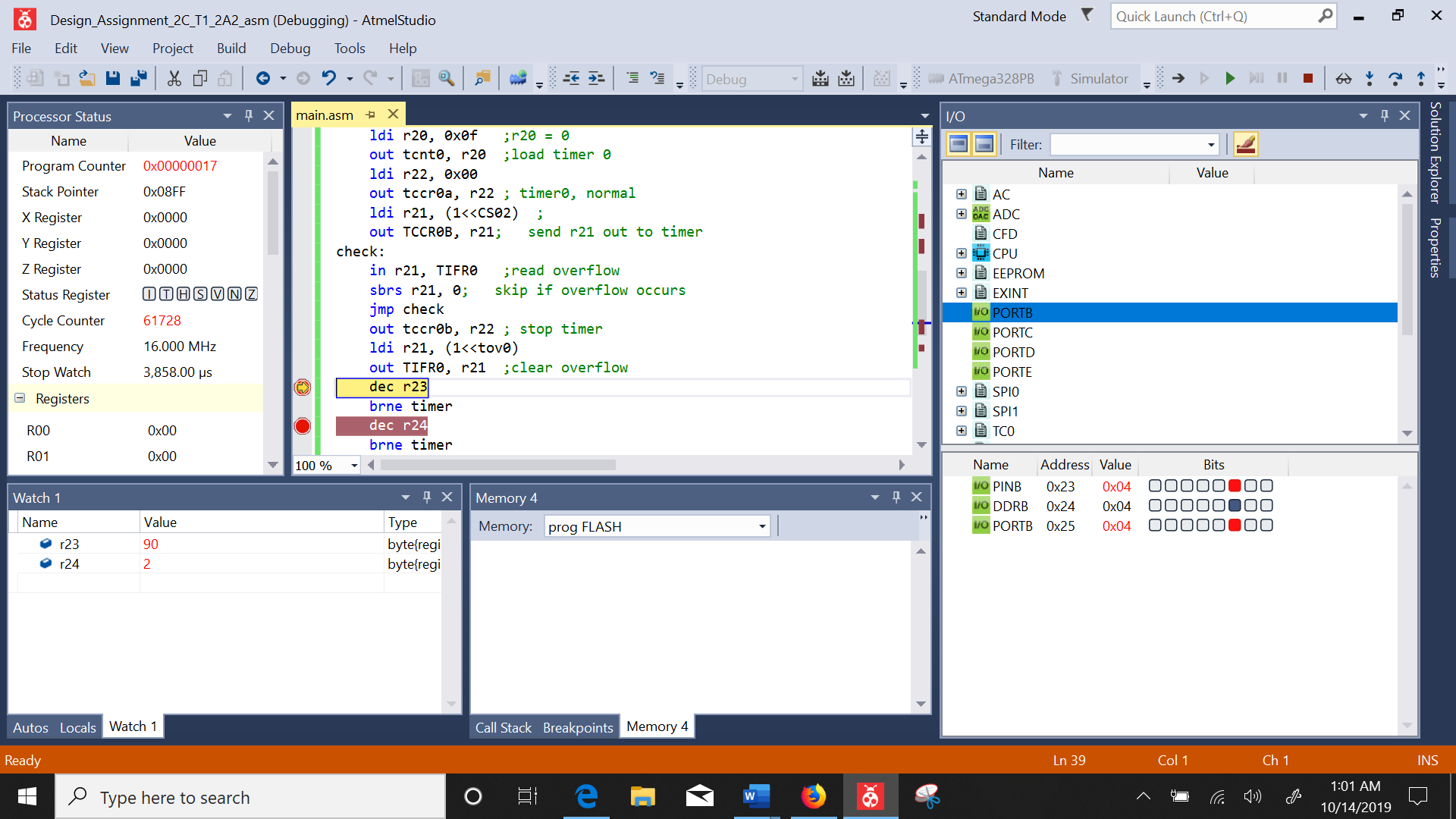


Figure : led on portb.2 is on

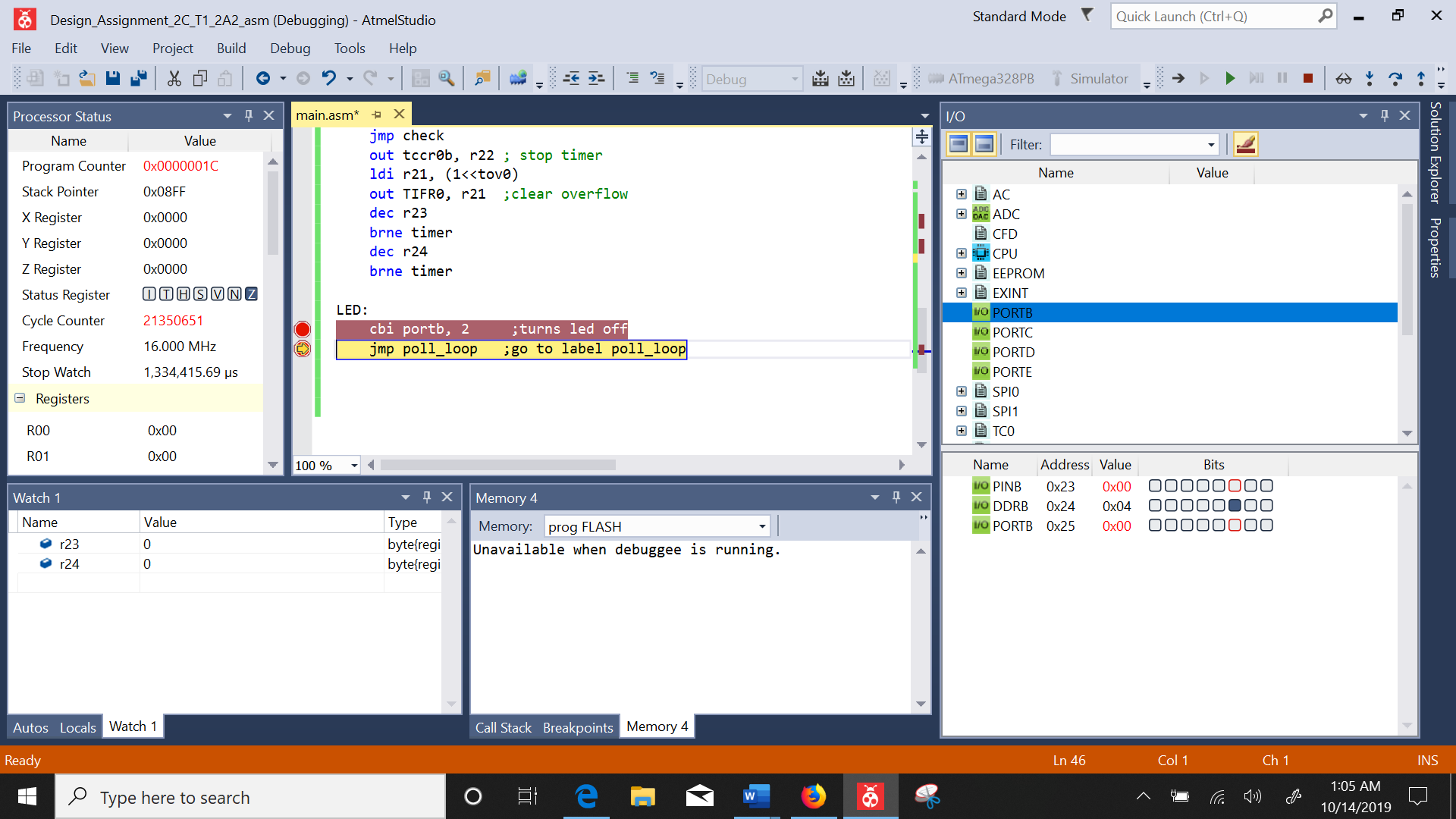


Figure : led turns off after 1.333s

TASK 2C\_T2\_2A.1:

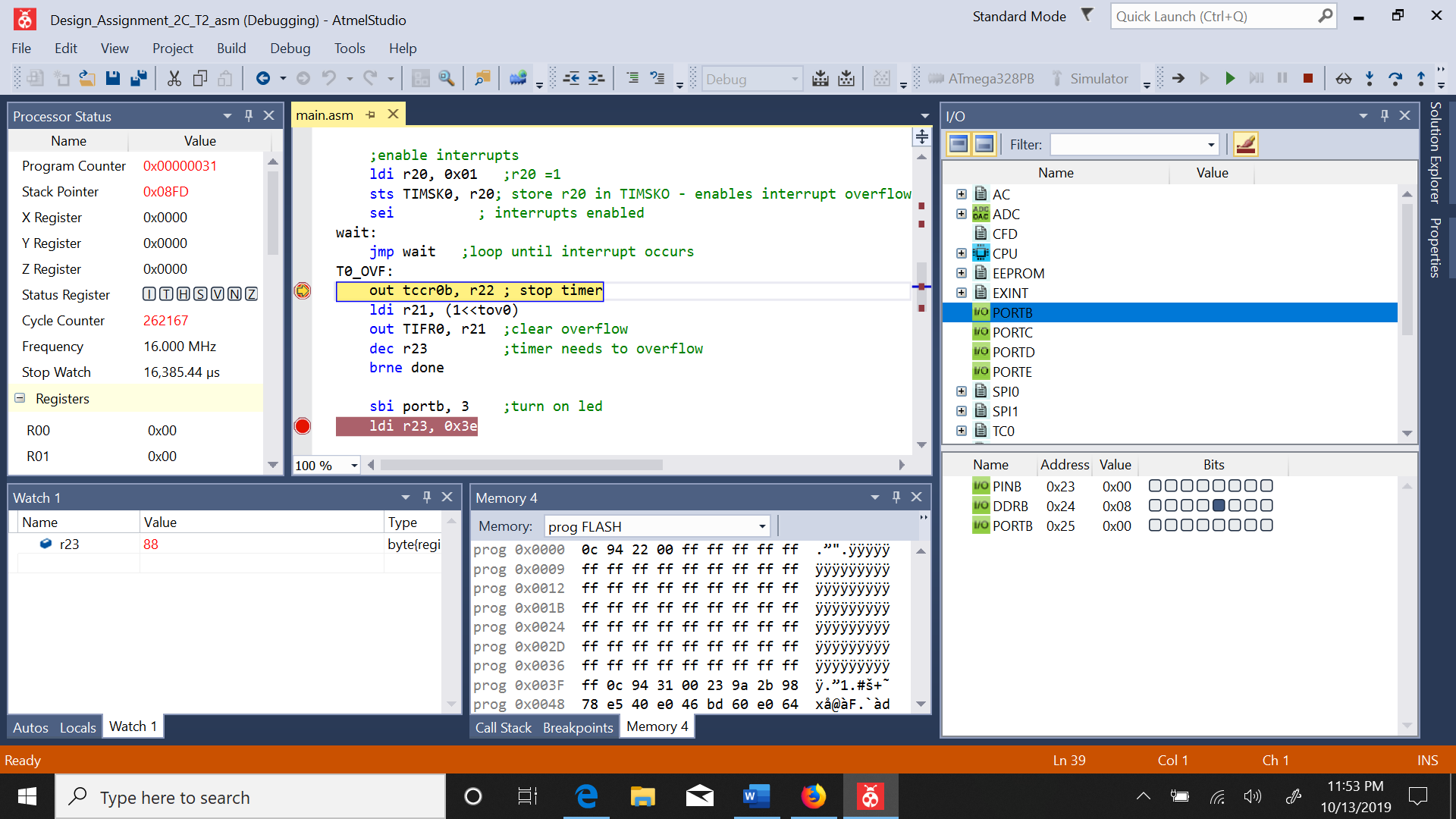


Figure : DDRB is set as output – LED is off

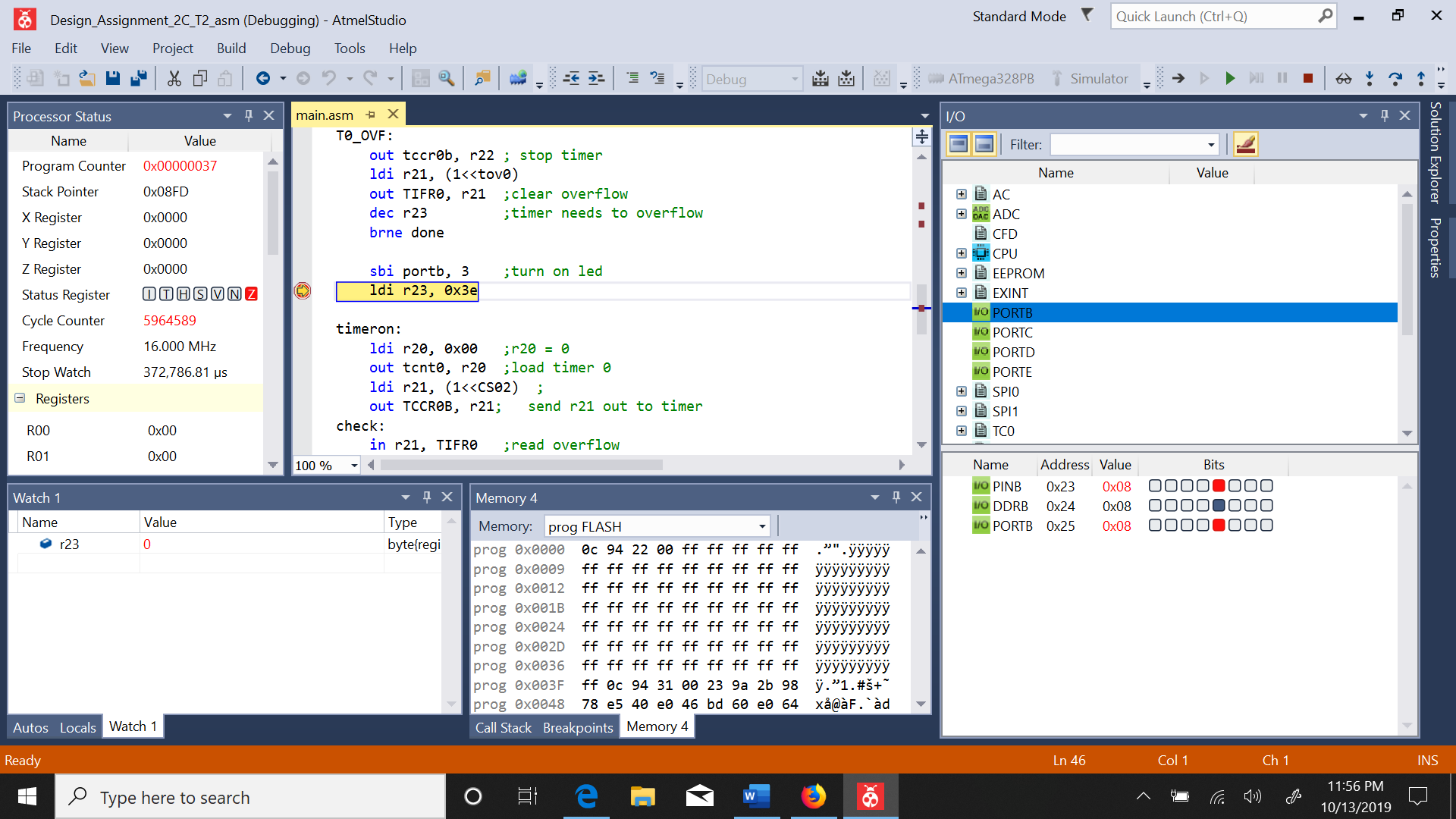


Figure : after delay of 372ms the led turns on

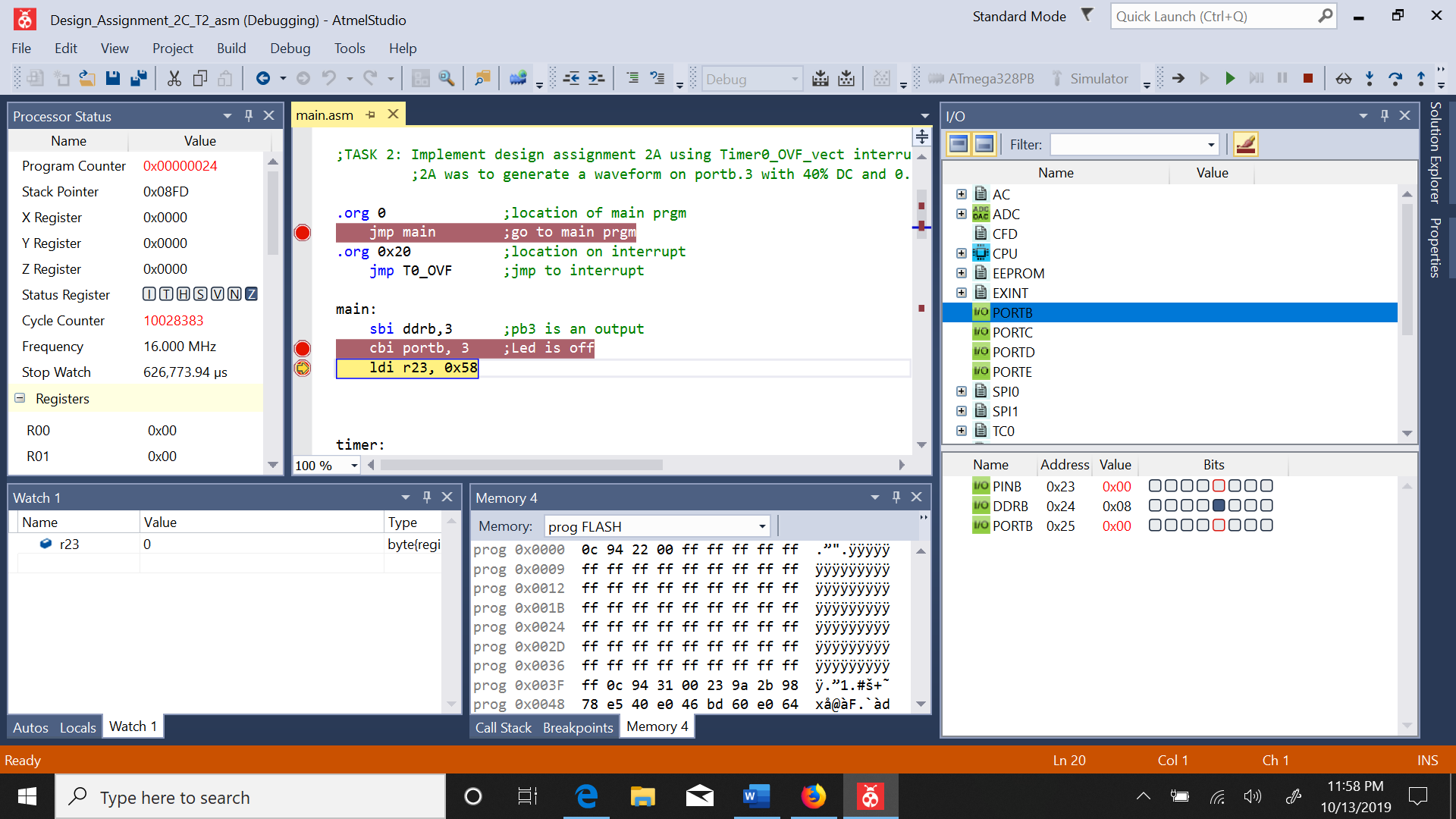


Figure : stopwatch is 626ms and led turns off

TASK 3.1:

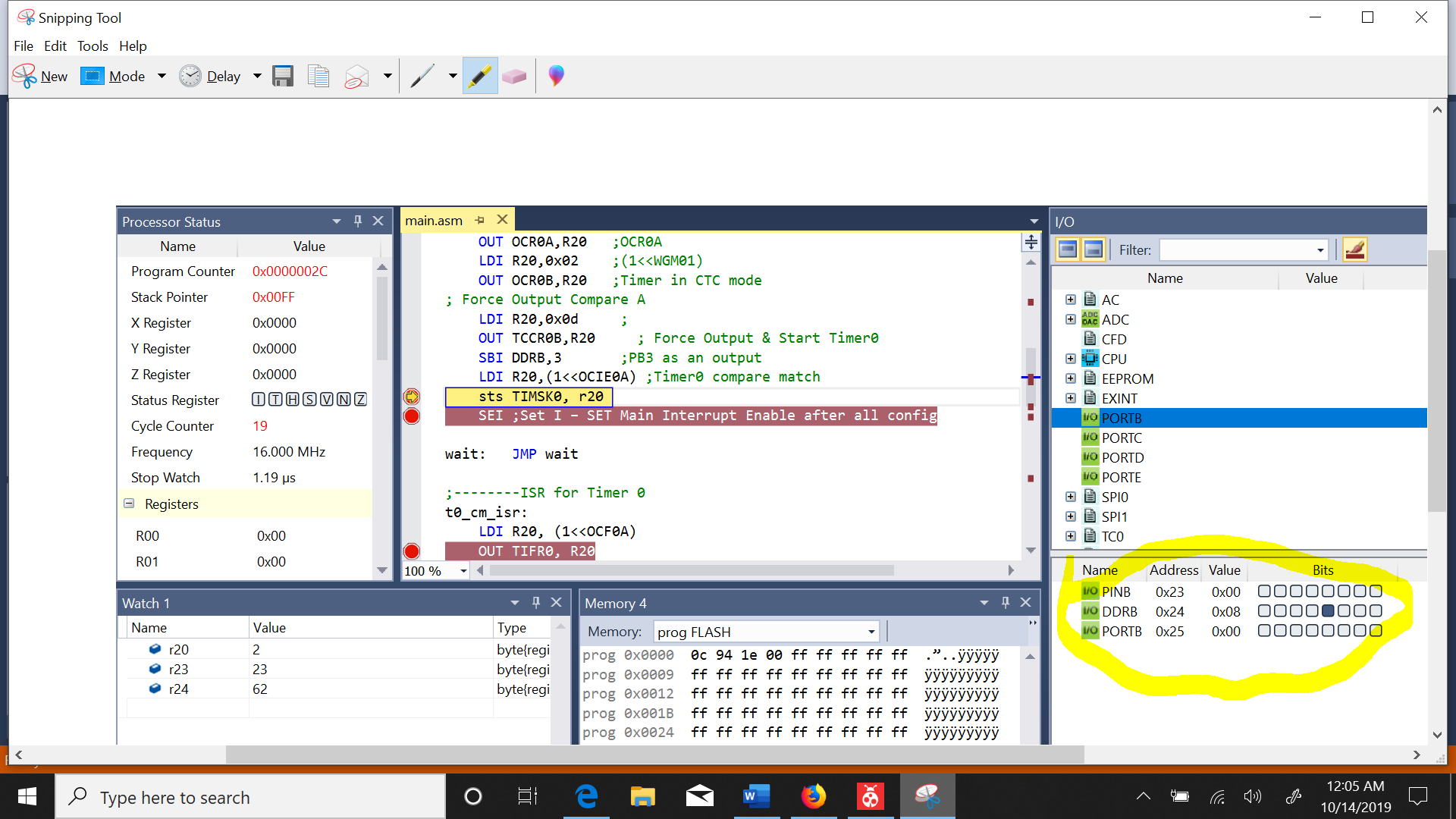


Figure : ddrb is output

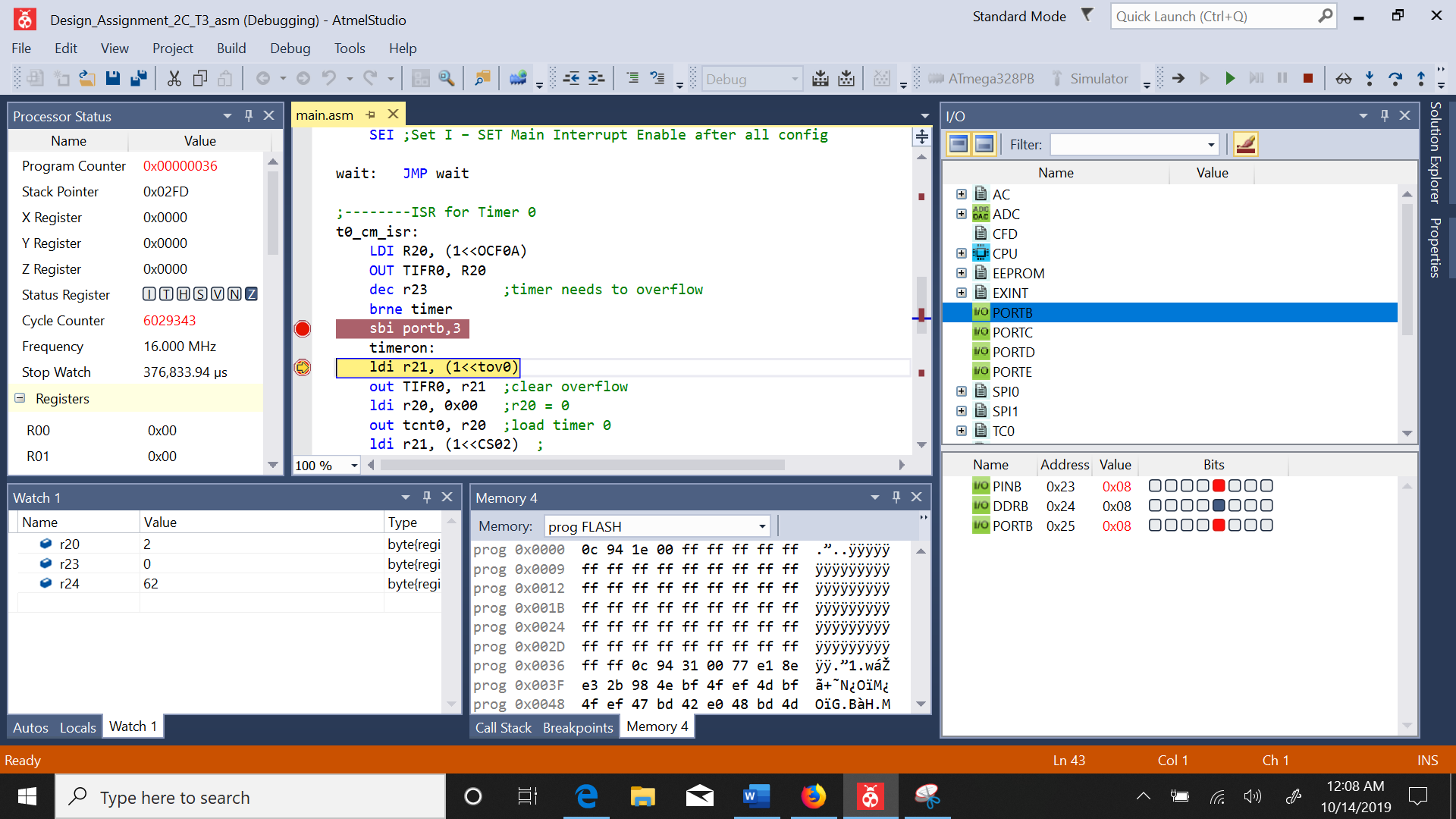


Figure : after 376ms LED turns on

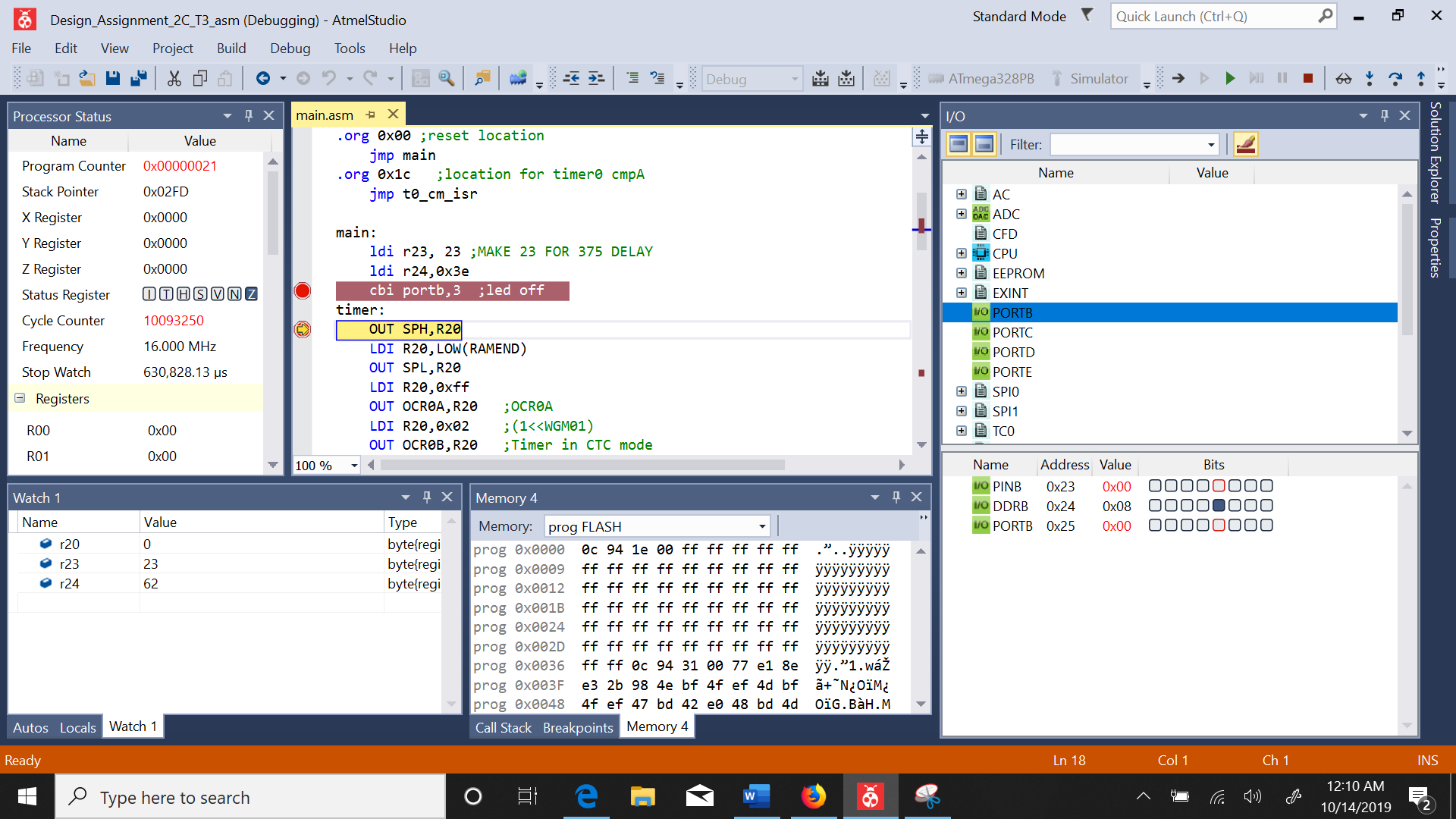


Figure : after 250ms the led turns off - total period is 630ms

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

Board was not set up due to lack of microcontroller

1. **VIDEO LINKS OF EACH DEMO**

There are no videos due to the lack of working microcontroller

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

NAME OF THE STUDENT